UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/082,591	02/22/2002	Ulfar Erlingsson	8828.0001-00	5689

22852 7590 03/28/2007

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

EXAMINER	
CAO, DIEM K	
ART UNIT	**PAPER NUMBER**
2194	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	03/28/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/082,591	ERLINGSSON, ULFAR	
	Examiner	Art Unit	
	Diem K. Cao	2194	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *22 December 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-47,49-62,64-99 and 101-114* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-47,49-62,64-99 and 101-114* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

WILLIAM THOMSON
SUPERVISORY PATENT EXAMINER

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-47, 49-62, 64-99 and 101-114 are pending. Applicant has amended claims 1, 49-52, 64, and 101-104.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-5, 8-9, 13-17, 19, 26-28, 30, 35-39, 40- 47, 49-54, 58-60, 64-69, 71, 78-80, 82, 87-91, 92-99, 101-106 and 110-112 are rejected under 35 U.S.C. 103(e) as being anticipated by Deianov et al. (U.S. 6,529,985 B1) in view of Wood et al (U.S. 2004/0210771 A1).

As to claim 1, Deianov teaches intercepting a service request (intercepting the system call; col. 3, lines 30-31 and col. 6, lines 28-38) made by a software component (processes 107; col. 6, lines 46-48 and col. 8, lines 15-16), evaluating the service request based on at least one rule (selectively intercept system calls; col. 6, lines 35-45), an original or modified data in the service request (process identifier; col. 7, lines 9-23. Notes that "or" is used, so meeting one of the two meet the limitation), and at least one of a present software system state (examiners the execution flag 131 ... executing; col. 8, lines 29-31) and a past software system state (the interception module ... wrapper 125; col. 8, lines 16-19), dynamically selecting at least one

desired behavior from among several behaviors for the software component based on the evaluation (execute the system call wrapper, or execute the system call 115 when the call was made by the wrapper; col. 8, lines 21-34), and dynamically controlling the software component such that the software component executes the desired behavior (execute system call wrapper or execute the actual system call; col. 8, lines 16-55).

Deianov does not teach dynamically alterable condition dependent rule. However, Wood teaches dynamically condition dependent rule (mapping of login credential types ... influenced by environment information such as time of request, source of request, connection speed; page 4, paragraph 37 and mapping rules may be dynamically varied ... effect; page 4, paragraph 38 and page 12, paragraph 87).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to apply the teaching of Wood to the system of Deianov because "dynamic alterable condition dependent rule" would provide the system of Deianov with the ability to allow access to the system can be changed based on the environment information in real time.

As to claim 2, Deianov teaches intercepting a service request comprises intercepting a software supported system call (system call; col. 3, lines 30-31 and col. 6, lines 28-38).

As to claim 3, Deianov teaches intercepting a software supported system call further comprises redirecting an entry in an interrupt vector table to alternative code (replace the pointer ... executes instead; col. 6, lines 21-27).

As to claim 4, Deianov teaches intercepting a hardware supported system call (access of the system hardware; col. 1, lines 31-40).

As to claim 5, Deianov teaches intercepting a hardware supported system call further comprises redirecting an entry in an interrupt vector table to alternative code (system call wrapper; col. 1, line 67 – col. 2, line 2 and col. 6, lines 21-27).

As to claim 8, Deianov teaches intercepting a subroutine based service (a system call is a subroutine; col. 1, lines 31-40).

As to claim 9, Deianov teaches intercepting a subroutine based service further comprises redirecting the subroutine call instruction to alternative code (system call wrapper; col. 1, line 67 – col. 2, line 2 and col. 6, lines 21-27).

As to claim 13, Deianov teaches executing alternative code in response to intercepting the service request (When a process makes a system call ... of the calling process; col. 8, lines 15-28).

As to claim 14, Deianov teaches executing alternative code in addition to calling the service request (When a process makes a system call ... of the calling process; col. 8, lines 15-28 and the system call was made by the wrapper; col. 8, lines 44-55).

As to claim 15, Deianov teaches the alternative code performs an operation with a same purpose as that of the service request (inherent from the wrapper makes the system call; col. 8, lines 44-55).

As to claim 16, Deianov teaches the alternative code performs an operation with a different purpose from that of the service request (inherent from the wrapper may or may not make the system call; col. 8, lines 44-55).

As to claim 17, Deianov teaches preventing execution of the service request (col. 2, lines 8-15).

As to claim 19, Deianov teaches preventing code that executes in response to interception of the service request from accessing at least some data (col. 2, lines 8-15).

As to claim 26, see rejection of claim 13 above.

As to claim 27, see rejection of claim 14 above.

As to claim 28, see rejection of claim 17 above.

As to claim 30, see rejection of claim 19 above.

As to claim 35, Deianov teaches preventing code that executes in response to interception of the service request from accessing a system resource (col. 2, lines 3-13).

As to claim 36, Deianov teaches the system resource comprises a network (opening a network communication channel; col. 1, lines 36-39).

As to claim 37, Deianov teaches the system resource comprises a storage media (reading data from a file; col. 1, lines 36-39).

As to claim 38, Deianov teaches the system resource comprises a file system (file system; col. 2, lines 9-10).

As to claim 39, Deianov teaches the system resource comprises a specific file (reading data from a file; col. 1, lines 36-39).

As to claim 40, Deianov does not teach the system resource comprises configuration information. Deianov teaches system calls can access system hardware or software resources, file system (col. 1, lines 31-39 and col. 2, lines 9-19). It would have been obvious that the system resource also include configuration information.

As to claim 41, Deianov does not teach the configuration information comprises registry data. However, it is well known in the art that the configuration information includes registry data

As to claim 42, Deianov teaches at least one condition dependent rule that specifies the desired behavior for the software component is static (list of selected process; col. 7, line 66 – col. 8, line 11).

As to claim 43, Deianov does not teach wherein dependant rules that specify the desired behavior for the software component comprise of static rules and dynamic rules. Woods teaches dependant rules that specify the desired behavior for the software component comprises a combination of static rules and dynamic rules (page 12, section 0087). It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of Deianov and Woods because it provides a method for improving the security of information transactions over networks.

As to claim 44, Deianov as modified teaches modifying the dynamic rules in response to behavior of the software component (col. 7, lines 29-32 and col. 9, lines 25-29).

As to claim 45, Deianov does not teach wherein modifying the dynamic rules further comprises responsive to an attempt by the software component to access specific data, creating a rule that specifies that the software component cannot access other data. Woods teaches

modifying the dynamic rules further comprises responsive to an attempt by the software component, by creating a rule that specifies whether the software component can or cannot access other data (page 4, section 0038).

As to claim 46, Deianov does not teach wherein modifying the dynamic rules further comprises responsive to an attempt by the software component to access specific data, creating a rule that specifies that the software component cannot perform certain functionality. Woods teaches modifying the dynamic rules further comprises responsive to an attempt by the software component, creating a rule that specifies whether the software component can or cannot perform certain functions (page 4, section 0038).

As to claim 47, Deianov teaches wherein the rules that specify the desired behavior for the software component are based on at least one of the following criteria: a user with which the software component is associated; identity of the software component; a time at which the software component is executing; history of the software component; a source of the software component; data which the software component attempts to access; functionality that software component attempts to execute; and computer network resources that the software component attempts to access (determining the calling process 107; col. 7, lines 7-23. It is noted that the reference needs only teach one of the above criteria).

As to claim 49, Deianov teaches an intercepting module (interception module), intercepting a service request made by a software component (intercepting the system call; col. 3,

lines 30-31 and col. 6, lines 28-38 and processes 107; col. 6, lines 46-48 and col. 8, lines 15-16), an altered states engine (interception module 111, initialization module 123; col. 7, lines 7-9), for evaluating the service request based on at least one rule (selectively intercept system calls; col. 6, lines 35-45), an original or modified data in the service request (process identifier; col. 7, lines 9-23. Notes that "or" is used, so meeting one of the two meet the limitation), and at least one of a present software system state (examiners the execution flag 131 ... executing; col. 8, lines 29-31) and a past software system state (the interception module ... wrapper 125; col. 8, lines 16-19), for dynamically selecting at least one desired behavior from among several behaviors for the software component (execute the system call wrapper, or execute the system call 115 when the call was made by the wrapper; col. 8, lines 21-34), alternative code for executing in response to an intercepted service request made by the software component (system call wrapper; col. 5, lines 52-53), for controlling the software component such that the software component executes the desired behavior (col. 8, lines 16-28), the alternative code being coupled to the altered states engine (col. 8, lines 56-58). Deianov further teaches the interception module provides the functionality of the intercepting module and the altered state engine (intercepting the system call; col. 3, lines 30-31 and interception module 111, initialization module 123; col. 7, lines 7-9). It would have been obvious to one of ordinary skill in the art that the interception module can be implemented as the intercepting module and the altered state engine for easier maintain.

Deianov does not teach dynamic alterable condition dependent rule, the altered states engine being coupled to the interception module, at least one rules database, for storing at least one dynamically alterable condition dependent rules, the rules database being coupled to the altered states engine. However, Wood teaches dynamically condition dependent rule (mapping of

login credential types ... influenced by environment information such as time of request, source of request, connection speed; page 4, paragraph 37 and mapping rules may be dynamically varied ... effect; page 4, paragraph 38 and page 12, paragraph 87), a table for storing static rule and a table for storing dynamic rules (mapping rules may be dynamically varied ... effect; page 4, paragraph 38 and page 12, paragraph 87).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to apply the teaching of Wood to the system of Deianov because "dynamic alterable condition dependent rule" would provide the system of Deianov with the ability to allow access to the system can be changed based on the environment information in real time, and improving the security of information transactions over networks.

As to computer system claim 50, it is the same as the method claim of claim 1 and is rejected under the same ground of rejection.

As to computer system claim 51, see rejection of claim 50 above.

As to claim 52, see rejection of claim 1 above. Deianov further teaches a computer readable medium on which the program codes are stored (inherent from "an interception module is loaded into the operating system"; col. 3, lines 34-36 and col. 5, line 63 – col. 6, line 4).

As to claim 53, see rejection of claim 13 above.

As to claim 54, see rejection of claim 19 above.

As to claims 58-60, see rejections of claims 26-28 above.

As to claim 64, see rejection of claim 1 above. Deianov further teach an altered state engine (The initialization module; col. 7, lines 30-36).

As to claims 65-69, see rejections of claims 13-17 above.

As to claim 71, see rejection of claim 19 above.

As to claims 78-80, see rejections of claims 26-28 above.

As to claim 82, see rejection of claim 30 above.

As to claims 87-94, see rejections of claims 35-42 above.

As to claims 95-98, see rejections of claims 43-46 above.

As to claims 99 and 101-106, see rejections of claims 47 and 49-54 above.

As to claims 110-112, see rejections of claims 58-60 above.

4. Claims 6-7 and 10-12 are rejected under 35 U.S.C. 103(a) as being unpatentable over Deianov et al. (U.S. 6,529,985 B1) in view of Wood et al (U.S. 2004/-210771 A1) further in view of APA (Admitted Prior Art).

As to claim 6, Deianov does not teach intercepting a service request comprises intercepting a software library based subroutine call. APA teaches intercepting a service request comprises intercepting a software library based subroutine call (page 9, lines 13-15). It would have been obvious to one of ordinary skill in the art to combine the teaching of Deianov and APA because this would improve the system of Deianov by allowing intercepting different type of service requests, not only the system calls.

As to claim 7, Deianov does not teach intercepting a software library based subroutine call further comprises modifying at least one dynamically linked library. APA teaches intercepting a software library based subroutine call further comprises modifying at least one dynamically linked library (page 9, lines 13-15).

As to claim 10, Deianov does not teach intercepting a subroutine base service further comprises patching machine language entry code of the subroutine. APA teaches intercepting a subroutine base service further comprises patching machine language entry code of the subroutine (page 9, lines 15-16).

As to claim 11, Deianov does not teach intercepting a service request comprises intercepting a service dispatch mechanism based on dynamic name resolution. APA teaches intercepting a service request comprises intercepting a service dispatch mechanism based on dynamic name resolution (page 9, lines 17-18).

As to claim 12, Deianov does not teach intercepting a service dispatch mechanism based on dynamic name resolution further comprises modifying service lookup name space. APA teaches intercepting a service dispatch mechanism based on dynamic name resolution further comprises modifying service lookup name space (page 9, lines 17-18).

5. Claims 20-23, 31-32, 55, 72-75, 83-84 and 107 are rejected under 35 U.S.C. 103(a) as being unpatentable over Deianov et al. (U.S. 6,529,985 B1) in view of Wood et al (U.S. 2004/-210771 A1) further in view of Chieu et al. (U.S. 6,587,888 B1).

As to claim 20, Deianov does not teach allowing code that executes in response to interception of the service request to access alternative data, different from requested data. Chieu teaches allowing code that executes in response to interception of the service request to access alternative data, different from requested data (col. 5, lines 41-43). It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of Deianov and Chieu because Chieu provides a method to implementing a dynamic software wrapper for discovery of non-exported functions and subsequent method interception.

As to claim 21, Deianov does not explicitly teach the alternative data comprises a copy of at least some data. Chieu teaches the alternate data comprises a copy of at least some data (col. 5, lines 41-43).

As to claim 22, Deianov teaches code that executes in response to the interception of the service request comprises at least alternative code (The interception module ... of the calling process; col. 8, lines 16-28).

As to claim 23, Deianov teaches code that executes in response to the interception of the service request comprises at least the service request (col. 8, lines 15-28 and lines 44-46).

As to claims 31-32, see rejections of claims 20-21 above.

As to claim 55, see rejection of claim 20 above.

As to claims 72-75, see rejections of claims 20-23 above.

As to claims 83-84, see rejections of claims 31-32 above.

As to claim 107, see rejections of claim 55 above.

6. Claims 18, 29, 61, 70, 81 and 113 are rejected under 35 U.S.C. 103(a) as being unpatentable over Deianov et al. (U.S. 6,529,985 B1) in view of Wood et al (U.S. 2004/-210771 A1) and Chieu et al. (U.S. 6,587,888 B1) further in view of Smale (U.S. 5,764,985).

As to claim 18, Deianov does note teach returning a value to the software component so as to simulate execution of the service request, without actually calling the service request. Smale teaches returning a value to the software component so as to simulate execution of the service request, without actually calling the service request (col. 5, lines 15-20). It would have been obvious to one of ordinary skill in the art to combine the teaching of Deianov and Smale because Smale provides a method to provide extended functionality to the lower level functions.

As to claim 29, see rejection of claim 18 above.

As to claim 61, see rejection of claim 29 above.

As to claim 70, see rejection of claim 18 above.

As to claim 81, see rejection of claim 29 above.

As to claim 113, see rejection of claim 61 above.

7. Claims 24-25, 56-57, 76-77 and 108-109 are rejected under 35 U.S.C. 103(a) as being unpatentable over Deianov et al. (U.S. 6,529,985 B1) in view of Wood et al (U.S. 2004/-210771 A1) further in view of Fin et al (U.S. 5,537,548).

As to claim 24, Deianov does not teach passing alternative parameters to code that executes in response to interception of the service request. Fin teaches passing alternative parameters to code that executes in response to interception of the service request (col. 4, lines 56-59). It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teaching of Deianov and Fin because Fin provides a method to modify the pass-in arguments to be processed by the new functions/codes.

As to claim 25, Fin teaches creating the alternative parameters by modifying original parameters passed to the service request (col. 4, lines 56-59).

As to claims 56-57, see rejections of claims 24-25 above.

As to claims 76-77, see rejections of claims 56-57 above.

As to claims 108-109, see rejections of claims 76-77 above.

8. Claims 33-34, 62, 85-86 and 114 are rejected under 35 U.S.C. 103(a) as being unpatentable over Deianov et al. (U.S. 6,529,985 B1) in view of Wood et al (U.S. 2004/-210771 A1) further in view of Smale (U.S. 5,764,985).

As to claim 33, Deianov does not teach returning an alternative value to the software component. Smale teaches returning an alternative value to the software component (col. 5, lines 12-20).

As to claim 34, Smale teaches creating the alternative value by modifying a value returned by the service request (col. 5, lines 12-20).

As to claim 62, see rejection of claim 33 above.

As to claims 85-86, see rejections of claims 33-34 above.

As to claims 114, see rejection of claim 62.

Response to Arguments

9. Applicant's arguments filed 12/22/2006 have been fully considered but they are not persuasive.

In the remarks, Applicant argued in substance that (1) Wood does not teach "dynamically alterable condition dependent rule" because in Wood, the mapping of trust levels to authentication schemes is static, i.e., a given trust level will always invoke the same

authentication scheme when using the existing mapping, and (2) no motivation to combine the teaching of Deianov and Wood.

Examiner respectfully disagrees with the Applicant's arguments:

- As to the point (1), Wood teaches the mapping rules is influenced by environment information such as time of request, connection speed, and/or client application environment (page 4, paragraph 37).
- As to the point (2), In response to applicant's argument that there is no suggestion to combine the references, the examiner recognizes that obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. See *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988)and *In re Jones*, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992). In this case, the motivation is found in the knowledge generally available to one of ordinary skill in the art.

Conclusion

10. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after

the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Diem K. Cao whose telephone number is (571) 272-3760. The examiner can normally be reached on Monday - Friday, 7:30AM - 3:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, William Thomson can be reached on (571) 272-3718. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Any response to this action should be mailed to:
Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist at 571-272-2100.

DC
March 20, 2007

WILLIAM THOMSON
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100